THE HONG KONG POLYTECHNIC UNIVERSITY
HONG KONG COMMUNITY COLLEGE

Subject Title	: Data Structures	**Subject Code**	: CCN2239
Session	: Semester One, 2013/14	**Time**	: 14:00 – 17:00
Date	: 20 December 2013	**Time Allowed**	: 3 Hours
Subject Examiner(s)	: Dr Pat CHAN Dr Joseph SO		

This question paper has a total of **NINETEEN** pages (including this covering page).

Instructions to Candidates:

1. There are THREE sections in this paper.
   - Section A (40%) – Multiple-choice Questions. Answer ALL questions in this section on the multiple-choice answer sheet provided. Each question carries 1 mark.
   - Section B (40%) – Short Questions. Answer any FOUR of the FIVE questions in this section in the answer book provided. Each question carries 10 marks.
   - Section C (20%) – Long Questions. Answer any ONE of the TWO questions in this section in the answer book provided. Each question carries 20 marks.
2. Candidates are NOT allowed to retain the multiple-choice answer sheet, the answer book and the examination question paper.
3. Show all your work clearly and neatly. Marks will be deducted for untidy work.
4. Reasonable steps should be shown.
5. All programming code must be written in Java programming language.

Authorised Materials:

	YES	NO
CALCULATOR	[ ]	[✓]
SPECIFICALLY PERMITTED ITEMS	[ ]	[✓]

DO NOT TURN OVER THE PAGE UNTIL YOU ARE TOLD TO DO SO

Section B (40%) – Short Questions

Answer any FOUR of the FIVE questions in this section in the answer book provided. Each question carries 10 marks. If more than FOUR questions are answered, only the first FOUR questions answered will be marked.

Question B1

You are given the following java program.

```
public class Triangle {

        public static void foo(int [] num) {
              for (int i = 0; i < num.length; i++)
                    System.out.print(num[i] + " ");
              System.out.println();
        }

        public static void fun1(int [] num) {
              for (int i = 0; i <= num.length; i++) {
                for (int j = num.length; j > i; j--) {
                    System.out.print("*");
                }
                for (int j = i-1; j > 1; j--) {
                    System.out.print(j + " ");
                }
                for (int j = 1; j < i; j++) {
                    num[j] += j;
                    System.out.print(j + " ");
                }
                System.out.println();
              }
        }

        public static void main(String [] args) {
              int [] num = { 9, 11, 8, 10, 2, 6 };
              System.out.print("The elements are: ");
              foo(num);

              fun1(num);

              System.out.println("After computing, the elements become:");
              foo(num);
        }
}
```

(a) What is the time complexity for the number of primitive operations by `num[j] += j;` in the `fun1()` method? Explain your answer. (5 marks)

(b) Show the output when successfully executing Triangle in java. (5 marks)

Question B2

(a) Given the following sort trace, identity which type of the probably sorting algorithm it is using with and explain your answers.

(i) Sorting Algorithm 1

(3 marks)

```
[ 13, 24, 14, 16, 60, 39, 8, 17, 50, 10]
[ 13, 14, 24, 16, 60, 39, 8, 17, 50, 10]
[ 13, 14, 16, 24, 60, 39, 8, 17, 50, 10]
[ 13, 14, 16, 24, 60, 39, 8, 17, 50, 10]
[ 13, 14, 16, 24, 39, 60, 8, 17, 50, 10]
[ 8, 13, 14, 16, 24, 39, 60, 17, 50, 10]
[ 8, 13, 14, 16, 17, 24, 39, 60, 50, 10]
[ 8, 13, 14, 16, 17, 24, 39, 50, 60, 10]
[ 8, 10, 13, 14, 16, 17, 24, 39, 50, 60]
```

(ii) Sorting Algorithm 2

(3 marks)

```
[ 30, 12, 64, 21, 46, 29, 21, 46, 68, 17, 11, 36, 9]
[ 30, 12, 64, 21, 46, 29, 21, 46, 68, 17, 36, 11, 9]
[ 30, 36, 64, 21, 46, 29, 21, 46, 68, 17, 12, 11, 9]
[ 30, 36, 64, 21, 46, 29, 21, 46, 68, 17, 12, 11, 9]
[ 30, 36, 64, 21, 46, 29, 68, 46, 21, 17, 12, 11, 9]
[ 30, 36, 64, 46, 46, 29, 68, 21, 21, 17, 12, 11, 9]
[ 30, 36, 64, 46, 46, 68, 29, 21, 21, 17, 12, 11, 9]
[ 68, 36, 64, 46, 46, 30, 29, 21, 21, 17, 12, 11, 9]
[ 68, 46, 64, 46, 36, 30, 29, 21, 21, 17, 12, 11, 9]
[ 68, 46, 64, 46, 36, 30, 29, 21, 21, 17, 12, 11, 9]
[ 68, 64, 46, 46, 36, 30, 29, 21, 21, 17, 12, 11, 9]
[ 68, 64, 46, 46, 36, 30, 29, 21, 21, 17, 12, 11, 9]
```

(b) Compare the bubble sort and the merge sort in terms of computational complexity.

(4 marks)

Question B3

Write a Java code segment to reverse the order of elements on Stack ***S***

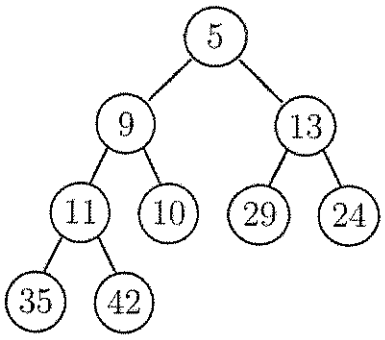
(a) using two additional stacks. (5 marks)

(b) using one additional queue. (5 marks)

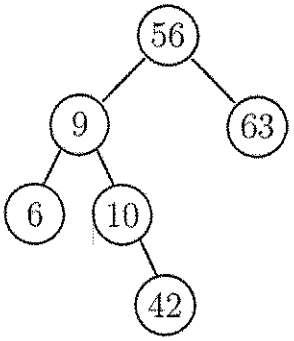
provided that the class *Stack* and *Queue* are well defined.

Question B4

(a) Given the following heap, draw the heap that would result after deleting the minimum element. (5 marks)



(b) The following tree was obtained by inserting the element 42 into an AVL tree. The tree no longer satisfies the AVL invariant, but the invariant can be re-established by performing two rotate operations. Show the tree after each of the rotate operations is done. (5 marks)



Question B5

(a) Recall that in a binary tree, a node may have 0, 1, or 2 children. In the following questions about binary trees, the *height* of a tree is the length (number of edges) of the longest path. A tree consisting of just one node has height 0.

(i) What is the maximum number of nodes in a binary tree of height d? (2 marks)

(ii) What is the minimum number of nodes in a binary tree of height d? (1 mark)

(iii) What is the maximum height of a binary tree containing n nodes? (2 marks)

(vi) What is the minimum height of a binary tree containing n nodes? (1 mark)

(b) What is the difference between binary tree and binary search tree? (4 marks)

- End of Section B -

Section C (20%) – Long Questions

Answer any ONE of the TWO questions in this section in the answer book provided. Each question carries 20 marks. If more than ONE question is answered, only the first ONE question answered will be marked.

Question C1

Given the following Java code:

```
import java.util.Random;

public class LinearList {
      private static Random rand1 = new Random();
      private static int [] list = {20, 36, 16, 36, 7, 48, 67};;

      public static void display() {
            if (list != null){
                  System.out.print("{" + list[0]);
                  for (int i = 1; i < list.length; i++)
                        System.out.print(", " + list[i]);
                  System.out.println("}");
            }
      }

      public static void randomPosIntArr(int n, int max) {
            if ( n<0 || max < 2) throw new IllegalArgumentException();
            int [] arr = new int[n];
            for ( int i=0; i <n; i++)
                  arr[i] = rand1.nextInt(max);

            list = arr;
      }
```

// code to be completed

```
      public static void main(String[] args) {

            LinearList.display();

            LinearList.reverse();
            LinearList.display();

            LinearList.truncate(3);
            LinearList.display();

            LinearList.randomPosIntArr(6, 60);
            LinearList.display();

            LinearList.uniqueRandomInts(10, 15);
            LinearList.display();

      }
}
```

Question C1 (continued)

(a) Linear list can be implemented by an array. Add the following static methods to perform the required functions for the list:

(i) `truncate(int n)` that shortens the list by keeping only the first n elements of `list[]`. (4 marks)

(ii) `reverse()` that reverses the order of the element in `list[]`. (4 marks)

(iii) `uniqueRandomInts(int n, int max)` that restructures `list[]` to become an array of length n with **unique** nonnegative random integers that are less than `max`. (8 marks)

(b) Show the output when successfully executing *LinearList* in java. (4 marks)

Note you may show only the added Java code in your answer.

Question C2

Given the following Java code:

```
public String choose(String[ ] data, int low, int high, int choice){
      if (low > high) {
            return data[low];
      }

      int mid = partition (data, low, high );

      if (choice<=mid) {
            return choose(data, low, mid-1, choice);
      }
      else {
            return choose(data, mid+1, high, choice);
      }
}

public void quickSort (String[] data, int low, int high){

      //code to be completed for part (c)

}

private int partition (String[] data, int low, int high){

      String pivot = data[low];
      int i = low;

      for(int j = low+1; j < high; j++){
            if(data[j].compareTo(pivot)<0){
                  i++;
                  String temp = data[i];
                  data[i] = data[j];
                  data[j] = temp;
            }
      }

      String temp = data[i];
      data[i] = data[low];
      data[low] = temp;

      return i;
}
```

Question C2 (continued)

(a) If the array `myData` contains the following strings, what value will be returned by `choose(myData, 0, 10, 7)`? (2 marks)

f	d	b	y	e	k	v	z	j	a
0	1	2	3	4	5	6	7	8	9

(b) Explain the function of the method `choose()` in general. (3 marks)

(c) Complete the specified `quicksort()` methods which implements the quick-sort algorithm for sorting a given string array. Note that you may use the methods provided in the above code and you may show only the added Java code in your answer. (6 marks)

(d) What is the worst-case input for quick-sort? Why? You must illustrate your answer with a brief example. (3 marks)

(e) Given the worst-case input described in (d), how will the insertion-sort versus merge-sort process be affected by this input? (6 marks)

- End of Section C -

- END OF PAPER -